Campionato Italiano di Salsa e Balli Caraibici

Documento Operazioni

SAPIENZA
UNIVERSITÀ DI ROMA

Studente: Simone Mille

Matricola: 1710762

Data di Consegna: 5/05/2021

Operazioni

Durante la fase di implementazione fisica e di progettazione dell'applicazione, sono state individuate altre operazioni da implementare, in aggiunta alle 12 presentate nel documento di analisi.
In questo documento vengono presentate in linguaggio naturale tutte le operazioni implementate nell'applicazione *Python*, la tabella delle frequenze e quella degli accessi, ed il codice SQL per ogni operazione.

Operazioni in Linguaggio Naturale

Operazione 1: Inserimento nuovo ballerino e relativa classe, categoria e scuola di ballo.
Operazione 2: Inserimento nuova scuola di ballo.
Operazione 3: Forma una nuova coppia assegnando ad essa un ballerino uomo ed una ballerina donna.
Operazione 4: Inserimento giudizio di un giudice relativo ad una *PartecipazioneC*.
Operazione 5: trova il piazzamento di un dato concorrente, in una specifica classifica.
Operazione 6: trova il numero dei ballerini appartenenti ad una certa scuola di ballo.
Operazione 7: trova i dati di tutti i giudici di un certo match.
Operazione 8: trova i dati di tutti i ballerini che compongono le coppie iscritte ad una determinata disciplina, ordinati per il codice del ballerino.
Operazione 9: visualizza nome, cognome e scuola di ballo dei ballerini di tutte le coppie arrivate in finale in una determinata classifica.
Operazione 10: calcola la percentuale del numero di ballerini di classe C, per scuola di ballo.
Operazione 11: memorizza l'iscrizione di un gruppo ad una disciplina.
Operazione 12: stampa tutti i dati di una disciplina di gruppo (incluso il numero di gruppi iscritti).
Operazione 13: estrai le scuole di ballo dove è massimo il numero di ballerini di una specifica classe.
Operazione 14: estrai per ogni coppia, il loro numero ed il numero di discipline a cui sono iscritte, ma solo se sono iscritte ad almeno 2 discipline.
Operazione 15: estrai Nome e Cognome dei giudici che hanno giudicato più match.
Operazione 16: estrai la classe a cui appartengono meno ballerini.
Operazione 17: estrarre il nome dei gruppi composti solo da ballerini dello stesso sesso.
Operazione 18: estrai nomi dei ballerini ed il numero di gruppi a cui partecipano, se quest'ultimo è maggiore di 1.
Operazione 19: Popola una specifica classifica.

Tabella delle Frequenze

Operazione	Tipo	Frequenza
Op.1	I	100 al mese
Op.2	I	16 al mese
Op.3	I	100 al mese
Op.4	I	1600 al giorno
Op.5	I	15 al mese
Op.6	I	10 al mese
Op.7	I	15 al mese
Op.8	B	5 al mese
Op.9	I	20 al mese
Op.10	B	5 al mese
Op.11	I	22 al mese
Op.12	I	20 al mese
Op.13	B	5 al mese
Op.14	B	5 al mese
Op.15	B	3 al mese
Op.16	B	3 al mese
Op.17	I	20 al mese
Op.18	I	20 al mese
Op.19	B	175 all’anno

Tabella degli Accessi e Codice SQL

Operazione 1: Inserimento nuovo ballerino e relativa classe, categoria e scuola di ballo.

Codice SQL:

```
INSERT INTO Ballerino (Nome, Cognome, Sesso, 'Data di Nascita', Privatista, Scuola, Classe, Categoria)
VALUES (?,?,?,?,?,?,?)
```

Oltre al codice SQL, è stato implementato anche un trigger, chiamato “ClasseScuola” che si occupa di aggiornare ScuolaDiBallo.

Concetto	Costrutto	Accessi	Tipo
Ballerino	Entità	1	S
ScuolaDiBallo	Entità	1	L
ScuolaDiBallo	Entità	1	S

Operazione 2: Inserimento nuova scuola di ballo.

Codice SQL:

```
INSERT INTO ScuolaDiBallo (Nome, Cap, Via, NumeroCivico)
VALUES (?,?,?,?)
```

Concetto	Accessi	Tipo
ScuolaDiBallo	1	S

Operazione 3: Forma una nuova coppia assegnando ad essa un ballerino uomo ed una ballerina donna.

Per implementare questa operazione, è stato necessario realizzare una stored procedure. Questo perché si ha la necessità di soddisfare il vincolo di derivazione RD1, che dice:
“La classe di un concorrente si ottiene calcolando qual’è la classe più alta dei ballerini che compongono il concorrente.”
Non era possibile farlo sfruttando la sintassi di SQL, perché le classi non seguono un ordine alfabetico; o meglio: lo seguono in parte. La classe M (Master) è la più alta, e da qui seguono le classi dalla A alla D in ordine alfabetico.

Inoltre, per come i dati della realtà di interesse sono stati rappresentati, l'inserimento di una coppia prescinde dall'inserimento precedente di un nuovo concorrente, rendendo di fatto l'operazione di inserimento di un concorrente ed il successivo inserimento di una coppia, un'operazione **atomica**. E' stato necessario quindi implementare questa operazione con una *transaction*.

Di seguito si riporta il codice SQL delle query utilizzate nella stored procedure, per il codice completo si fa riferimento alla funzione "query3" del file sql.py.

Disclaimer: si noti che, per ragioni di lettura del codice, alcune SELECT non sono state usate come sottoquery delle insert, si è invece provveduto ad effettuarle prima e a salvare i valori in delle variabili, usate poi come VALUES delle operazioni di inserimento. Sebbene questo sia normalmente inefficiente a causa dei costi di comunicazione, essendo SQLite un DB *server-less*, è possibile comunicare tra applicazione e database ad un costo irrisorio.

Codice SQL:

```
SELECT Categoria
FROM Ballerino
WHERE Codice = ballerino

SELECT Classe
FROM Ballerino
WHERE Codice = dancer

BEGIN

INSERT INTO Concorrente (Classe, Categoria)
VALUES (?,?)

SELECT max(Numero)
FROM Concorrente

INSERT INTO Coppia (Numero, Ballerino, Ballerina)
VALUES (?, ?, ?)

COMMIT
```

Concetto	Accessi	Tipo
Ballerino	3	L
Concorrente	1	S
Concorrente	(1, 200)	L
Coppia	1	S

Gli accessi sono tutti self-explanatory, apparte per gli accessi in lettura a concorrente. Il valore (1, 200), deriva da quanti concorrenti sono già stati inseriti nel database, 1 nel caso migliore, 200 (ovvero tutti, come indicato nella tabella dei volumi) nel caso peggiore.

Operazione 4: Inserimento giudizio di un giudice relativo ad una *Partecipazione - C*.

Per implementare questa operazione, c'è stato bisogno di realizzare una stored procedure, in maniera tale da delegare all'applicazione il controllo di alcuni elementi.
In particolare, per soddisfare il vincolo RV5, sono stati introdotti dei controlli che verificassero che il giudice non avesse esaurito i "Si" a sua disposizione. Questo ha portato ad un numero di accessi maggiore rispetto a quelli preventivati in precedenza.
Di seguito si riporta il codice SQL delle query utilizzate nella stored procedure, per il codice completo si fa riferimento alla funzione "query4" del file sql.py.

Codice SQL:

```
SELECT Round FROM Match
WHERE ID = match

SELECT count(*)
FROM Giudizio
WHERE PartecipazioneG = part_g and MatchG = match

INSERT INTO Giudizio (PartecipazioneC, PartecipazioneG, MatchC, MatchG, Tipologia,
Consenso)
VALUES (?, ?, ?, ?, ?, ?)
```

Concetto	Accessi	Tipo
Match	1	L
Giudizio	14	L
Giudizio	1	S

Gli accessi in lettura di Giudizio sono 14, poiché al caso pessimo un giudice può aver già dato 14 "Si".

Operazione 5: trova il piazzamento di un dato concorrente, in una specifica classifica.

Codice SQL:

```
SELECT Posizione
```

```
FROM Piazzamento join Concorrente on Concorrente = Numero
WHERE Numero = concorrente and Classifica in
( SELECT ID
FROM Classifica
WHERE Classe =  classe and Categoria = categoria and Disciplina = disciplina
)
```

Concetto	Accessi	Tipo
Classifica	1	L
Concorrente	1	L
Piazzamento	1	L

Operazione 6: trova il numero dei ballerini appartenenti ad una certa scuola di ballo.

Codice SQL:

```
SELECT (NumBalleriniD + NumBalleriniC + NumBalleriniB + NumBalleriniA +
NumBalleriniM) as Totale
FROM ScuolaDiBallo
WHERE  ScuolaDiBallo.NumeroCivico =  civ  and ScuolaDiBallo.Via = via and
ScuolaDiBallo.Cap = cap
```

Concetto	Accessi	Tipo
ScuolaDiBallo	1	L

Operazione 7: trova i dati di tutti i giudici di un certo match.

Codice SQL:

```
select *
from Giudice as G
where Codice in
(
   select Giudice
   from PartecipazioneG
   where Match in
   (
         select ID
         from Match
```

where Round = m_roundand Classe = m_classe and Categoria = m_categoria and Disciplina = m_disciplina
)

Concetto	Accessi	Tipo
Match	1	L
PartecipazioneG	6	L
Giudice	6	L

Operazione 8: trova i dati di tutti i ballerini che compongono le coppie iscritte ad una determinata disciplina, ordinati per il codice del ballerino.

Codice SQL:

```
select *
from Ballerino
 where (Sesso = "Femmina" and Codice in
        (
                select Ballerina
                from Coppia
                where Numero in
               (
                        select NumeroCoppia
                        from  IscrizioneCoppia
                        where NomeDisciplina = disciplina
               )
        ))
        or
        Sesso = "Maschio" and Codice in
        (
                select Ballerino
                from Coppia
                where Numero in
                (
                        select NumeroCoppia
                        from IscrizioneCoppia
                        where NomeDisciplina = disciplina
                )
        )
  order by Codice
```

Concetto	Accessi	Tipo

IscrizioneCoppia	100	L
Coppia	100	L
Ballerino	200	L

Operazione 9: visualizza nome, cognome e scuola di ballo dei ballerini di tutte le coppie arrivate in finale in una determinata classifica.

Codice SQL:

```
select B1.Nome, B1.Cognome, B1.Scuola, B2.Nome as NomeF, B2.Cognome as CognomeF, B2.Scuola as ScuolaF, C.Numero
from  Ballerino as B1 join Coppia as C on Ballerino = B1.Codice
        join Ballerino as B2 on Ballerina = B2.Codice
 group by C.Numero
 having C.Numero in
        (
                select  Concorrente
                from Piazzamento join Classifica as C on Piazzamento.Classifica = C.ID
                 where Classe = classe and Categoria = categoria and Disciplina = disciplina
        )
```

Concetto	**Accessi**	**Tipo**
Classifica	1	L
Piazzamento	6	L
Coppia	6	L
Ballerino	12	L

Operazione 10: calcola la percentuale del numero di ballerini di classe C, per scuola di ballo.

Codice SQL:

```
select ID, Nome,
(NumBalleriniC/(NumBalleriniD+NumBalleriniC+NumBalleriniB+NumBalleriniA+NumBallerini
M)) as Percentuale
from ScuolaDiBallo
```

Concetto	**Accessi**	**Tipo**

ScuolaDiBallo	25	L

Operazione 11: memorizza l'iscrizione di un gruppo ad una disciplina.

Codice SQL:

```
INSERT INTO IscrizioneGruppo
VALUES (?, ?)
```

Concetto	Accessi	Tipo
IscrizioneGruppo	1	S
Disciplina	1	L
Disciplina	1	S

Gli accessi alla relazione "Disciplina" sono dovuti al *trigger* "UpdateNumeroGruppi" che si attiva dopo un'insert di IscrizioneGruppo.

Operazione 12: stampa tutti i dati di una disciplina di gruppo (incluso il numero di gruppi iscritti).

Codice SQL:

```
select *
from Disciplina
where Nome = disciplina
```

Concetto	Accessi	Tipo
Disciplina	1	L

Operazione 13: estrai le scuole di ballo dove è massimo il numero di ballerini di una specifica classe.

Codice SQL:

```
select *
```

```
from ScuolaDiBallo
group by ID
having max(NumBalleriniX)
```

Concetto	Accessi	Tipo
ScuolaDiBallo	25	L

Operazione 14: estrai per ogni coppia, il loro numero ed il numero di discipline a cui sono iscritte, ma solo se sono iscritte ad almeno 2 discipline.

Codice SQL:

```
select C.Numero, count(IC.NomeDisciplina) as N°_Discipline
from Coppia as C join IscrizioneCoppia as IC on C.Numero = IC.NumeroCoppia
group by C.Numero
having count(IC.NomeDisciplina) > 1
```

Concetto	Accessi	Tipo
Coppia	150	L
IscrizioneCoppia	300	L

Operazione 15: estrai Nome e Cognome dei giudici che hanno giudicato più match.

Codice SQL:

```
select G.Nome, G.Cognome
from Giudice as G join PartecipazioneG as PG on G.Codice = PG.giudice
group by G.Codice
having count(*) =
        (
                select max(c)
                from (
                        select count(*) as c
                        from Giudice as G join PartecipazioneG as PG on G.Codice =
PG.giudice
                        group by G.Codice
                        )
        )
```

Concetto	Accessi	Tipo
Giudice	20	L
PartecipazioneG	1800	L

Operazione 16: estrai la classe a cui appartengono meno ballerini.

Codice SQL:

```
select C.Nome
from Classe as C join Ballerino as B on C.Nome = B.Classe
group by C.Nome
having count(*) =
        (
                select min(c)
                from (
                        select count(*) as c
                        from Classe as C join Ballerino as B on C.Nome = B.Classe
                        group by C.Nome
                        )
        )
```

Concetto	Accessi	Tipo
Classe	5	L
Ballerino	300	L

Operazione 17: estrarre il nome dei gruppi composti solo da ballerini dello stesso sesso.

Codice SQL:

```
select Gruppo.Nome
from Gruppo join ComposizioneG as CG on Gruppo.Numero = CG.Gruppo
group by Gruppo.Nome
having count(*) =
        (
```

```
        select count(*)
        from Ballerino as B join ComposizioneG as CG on B.Codice = CG.Ballerino
        where B.Sesso = 'Maschio' and CG.Gruppo = Gruppo.Numero
)
or
count (*) =
(
        select count(*)
        from Ballerino as B join ComposizioneG as CG on B.Codice = CG.Ballerino
        where B.Sesso = 'Femmina' and CG.Gruppo = Gruppo.Numero
)
```

Concetto	Accessi	Tipo
Gruppo	50	L
ComposizioneG	325	L
Ballerino	325	L

Operazione 18: estrai nomi dei ballerini ed il numero di gruppi a cui partecipano, se quest'ultimo è maggiore di 1.

Codice SQL:

```
select B.Nome, B.Cognome, count(*) as Num
from Ballerino as B join ComposizioneG as CG on B.Codice = CG.Ballerino
group by B.Codice
having 1 < count(*)
```

Concetto	Accessi	Tipo
Ballerino	300	L
ComposizioneG	325	L

Operazione 19: Popola una specifica classifica.

Per implementare questa operazione, è stata creata una stored procedure.
Come si evince dal vincolo di derivazione RD2:

"I primi 6 piazzamenti di una classifica si ottengono calcolando, per ogni concorrente

partecipante alla finale, la media dei piazzamenti assegnati dai 6 giudici. I concorrenti vengono quindi classificati in ordine crescente. Chi risulta avere la media più bassa è il primo, chi la ha più alta è il sesto. E' possibile che due o più concorrenti abbiano la media uguale, in quel caso il loro piazzamento sarà uguale."

Per il codice completo della stored procedure, si faccia riferimento alla funzione query19 del file sql.py.
Di seguito vengono presentate le interrogazioni SQL utilizzate nella stored procedure. Per ogni interrogazione, viene presentata una tabella degli accessi.

Codice SQL:

Recuperiamo il numero di concorrenti che hanno partecipato alla finale:

```
SELECT count(*)
FROM Match as M join PartecipazioneC as PC on M.ID = PC.Match
WHERE M.Classe = classe and M.Categoria = categoria and M.Disciplina = disciplina and
M.Round = Finale
```

Concetto	Accessi	Tipo
Match	1	L
PartecipazioneC	6	L

Recuperiamo il numero dei giudizi relativi a quel match:

```
SELECT count(*)
FROM Match as M join PartecipazioneC as PC on M.ID = PC.Match
        join Giudizio as G on G.PartecipazioneC = PC.Concorrente and M.ID = G.MatchC
WHERE M.Classe = classe and M.Categoria = categoria and M.Disciplina = disciplina and
M.Round = Finale
```

Concetto	Accessi	Tipo
Match	1	L
PartecipazioneC	6	L
Giudizio	36	L

Recuperiamo l'id della Classifica scelta dall'utente.

```
SELECT ID
FROM Classifica as M
WHERE M.Classe = classe and M.Categoria = categoria and M.Disciplina = disciplina
```

Concetto	Accessi	Tipo
Classifica	1	L

Per ogni concorrente si calcola la media dei giudizi, e si ordina il tutto per media.

```
SELECT PC.Concorrente, avg(G.Posizione) as Media
FROM Match as M join PartecipazioneC as PC on M.ID = PC.Match
        join Giudizio as G on G.PartecipazioneC = PC.Concorrente
WHERE M.Classe = classe and M.Categoria = categoria and M.Disciplina = disciplina and
M.Round = Finale
GROUP BY PC.Concorrente
ORDER BY avg(G.Posizione)
```

Concetto	Accessi	Tipo
Match	1	L
PartecipazioneC	6	L
Giudizio	36	L

Si eseguono gli inserimenti. Attenzione: il numero di inserimenti è uguale al numero di concorrenti della finale.

```
INSERT INTO Piazzamento
VALUES (?, ?, ?)
```

Concetto	Accessi	Tipo
Piazzamento	6	S

OPERAZIONI SQL CON VISTE

Come da requisiti, alcune delle operazioni sono state implementate **anche** con l'ausilio di viste.
Di seguito viene presentato il codice SQL.

Operazione 9: visualizza nome, cognome e scuola di ballo dei ballerini di tutte le coppie arrivate in finale in una determinata classifica.

```
create view Coppie as
select B1.Nome as B1Nome, B1.Cognome as B1Cognome, B1.Scuola as B1Scuola,
B2.Nome as B2Nome, B2.Cognome as B2Cognome, B2.Scuola as B2Scuola, C.Numero as
Numero
from Ballerino as B1 join Coppia as C on Ballerino = B1.Codice
        join Ballerino as B2 on Ballerina = B2.Codice
group by C.Numero
```

```
select B1Nome, B1Cognome, B1Scuola, B2Nome, B2Cognome, B2Scuola, Numero
from Coppie
where Numero in
(
        select Concorrente
        from Piazzamento join Classifica as C on Piazzamento.Classifica = C.ID
        where Classe = 'M' and Categoria = 'Adulti' and Disciplina = 'Salsa'
)
```

Operazione 14: estrai per ogni coppia, il loro numero ed il numero di discipline a cui sono iscritte, ma solo se sono iscritte ad almeno 2 discipline.

```
create view Coppie_Discipline as
select C.Numero as Numero, count(IC.NomeDisciplina) as N°_Discipline
from Coppia as C join IscrizioneCoppia as IC on C.Numero = IC.NumeroCoppia
group by C.Numero
```

```
select Numero, N°_Discipline
from Coppie_Discipline
where N°_Discipline > 1
```

Operazione 15: estrai Nome e Cognome dei giudici che hanno giudicato più match.

```
create view Giudici_Match as
select G.Nome as Nome, G.Cognome as Cognome, count(*) as Num
from Giudice as G join PartecipazioneG as PG on G.Codice = PG.giudice
group by G.Codice


select Nome, Cognome, Num
from Giudici_Match
where Num =
(
        select max(num)
        from Giudici_Match
)
```

Operazione 16: estrai la classe a cui appartengono meno ballerini.

```
create view Ballerini_Classe as
select C.Nome as Nome, count(*) as Num
from Classe as C join Ballerino as B on C.Nome = B.Classe
group by C.Nome

select Nome, Num
from Ballerini_Classe
where Num =
(
        select min(num)
        from Ballerini_Classe
)
```

Operazione 18: estrai nomi dei ballerini ed il numero di gruppi a cui partecipano, se quest'ultimo è maggiore di 1.

```
create view Gruppi as
select B.Nome, B.Cognome, count(*) as Num
from Ballerino as B join ComposizioneG as CG on B.Codice = CG.Ballerino
group by B.Codice

select Nome, Cognome, Num
from Gruppi
where Num > 1
```